import java.util.ArrayList;

import java.util.Arrays;

import java.util.Objects;

import java.util.\*;

public class Poligoni {

public static void main(String[] args) {

ArrayList<Poligono> a = new ArrayList<>();

Quadrato q1 = new Quadrato(5.0);

Quadrato q2 = new Quadrato(2.0);

Quadrato q3 = new Quadrato(6.0);

Rettangolo r1 = new Rettangolo(2.0, 3.0);

Rettangolo r2 = new Rettangolo(3.0, 5.0);

TriangoloRettangolo tr1 = new TriangoloRettangolo(3.0, 4.0);

a.add(q1);

a.add(r1);

a.add(q2);

a.add(q3);

a.add(r2);

a.add(tr1);

System.out.println("Elenco non ordinato:");

for (Poligono p : a) {

System.out.println(p + " AREA: " + p.area() + " PERIMETRO: " + p.perimetro());

}

Collections.sort(a);

System.out.println("\nElenco in ordine 'naturale':");

for (Poligono p : a) {

System.out.println(p + " AREA: " + p.area() + " PERIMETRO: " + p.perimetro());

}

PoligonoComparatorePerArea pca = new PoligonoComparatorePerArea();

Collections.sort(a, pca);

System.out.println("\nElenco in ordine di area:");

for (Poligono p : a) {

System.out.println(p + " AREA: " + p.area() + " PERIMETRO: " + p.perimetro());

}

}

}

//---------------------------------------------------------------------------------------

abstract class Poligono implements Comparable<Poligono> {

protected String nome;

protected int numLati;

protected double[] lati;

public Poligono() {

this.nome = "";

this.numLati = 0;

this.lati = null;

}

public Poligono(String nome, double[] lati) {

this.nome = nome;

this.numLati = lati.length;

this.lati = lati;

}

public double[] getLati() {

return lati;

}

public String getNome() {

return nome;

}

public int getNumLati() {

return numLati;

}

public double perimetro() {

double p = 0;

for (double lato : lati) {

p += lato;

}

return p;

}

public abstract double area();

@Override

public boolean equals(Object obj) {

if (this == obj) {

return true;

}

if (obj == null) {

return false;

}

if (getClass() != obj.getClass()) {

return false;

}

final Poligono other = (Poligono) obj;

if (!Objects.equals(this.nome, other.nome)) {

return false;

}

return Arrays.equals(this.lati, other.lati);

}

@Override

public int compareTo(Poligono altro) {

return this.nome.compareTo(altro.nome);

}

@Override

public String toString() {

return "Poligono{" + "nome=" + nome + ", numLati=" + numLati + ", lati=" + Arrays.toString(lati) + '}';

}

}

//---------------------------------------------------------------------------------------

class Quadrato extends Poligono {

public Quadrato(double lato) {

nome = "Quadrato";

lati = new double[4];

numLati = lati.length;

lati[0] = lato;

lati[1] = lato;

lati[2] = lato;

lati[3] = lato;

}

@Override

public double area() {

return lati[0] \* lati[0];

}

}

//---------------------------------------------------------------------------------------

class Rettangolo extends Poligono {

public Rettangolo(double base, double altezza) {

nome = "Rettangolo";

lati = new double[4];

numLati = lati.length;

lati[0] = base;

lati[1] = altezza;

lati[2] = base;

lati[3] = altezza;

}

@Override

public double area() {

return lati[0] \* lati[1];

}

}

//----------------------------------------------------------------------------------------

class TriangoloRettangolo extends Poligono {

public TriangoloRettangolo(double cat1, double cat2) {

nome = "Triangolo rettangolo";

lati = new double[3];

numLati = lati.length;

lati[0] = cat1;

lati[1] = cat2;

lati[2] = Math.sqrt(cat1 \* cat1 + cat2 \* cat2); // ipotenusa

}

@Override

public double area() {

return lati[0] \* lati[1] / 2.0;

}

}

//---------------------------------------------------------------------------------------

class PoligonoComparatorePerArea implements Comparator<Poligono> {

@Override

public int compare(Poligono a, Poligono b) {

return Double.compare(a.area(), b.area());

}

}